PATENT APPLICATION
27600/M195A

APPENDIX

Script Of: application "DB Tool Combos v2.3 source"

```
global imfile
--global btfile2
global writestr
global keywordlist
global skukeylist
global fn
global mygotimage
global maincatlist
global subcatlist
global detailcatlist
global detailcatassoc
global excelfile_to_open
global currentitemnumber
global currentskunumber
global issku
global firstkeyword
global firstdescription
global combos
global stylelist
global colorlist
global sizelist

set mygotimage to false
set skumods to 0
set fn to ""
set currentitemnumber to 2
set currentskunumber to 2
set currentimagenumber to 2
set issku to {}
set firstkeyword to true
set firstdescription to true
set colorlist to {}
set sizelist to {}
set stylelist to {}

tell application "Microsoft Excel"
 Activate
 set excelfile_to_open to GetOpenFilename Title "Select a Blank Excel Template File?"
 Open excelfile_to_open

 tell Worksheet "Products"
  set notfound to true
  repeat while notfound
   set cellstring to ("r" & currentitemnumber as text) & "c1"
   set cellc to (the Value of Cell cellstring)
   if (cellc = 0) then
```

Script Of: application "DB Tool Combos v2.3 source"

```
   set notfound to false
   else
   set currentitemnumber to currentitemnumber + 1
   end if
  end repeat
 end tell

 tell Worksheet "Skus"
  set notfound to true
  repeat while notfound
   set cellstring to ("r" & currentskunumber as text) & "c1"
   set cellc to (the Value of Cell cellstring)
   if (cellc = 0) then
    set notfound to false
   else
    set currentskunumber to currentskunumber + 1
   end if
  end repeat
 end tell

 tell Worksheet "Alternate Image"
  set notfound to true
  repeat while notfound
   set cellstring to ("r" & currentimagenumber as text) & "c1"
   set cellc to (the Value of Cell cellstring)
   if (cellc = 0) then
    set notfound to false
   else
    set currentimagenumber to currentimagenumber + 1
   end if
  end repeat
 end tell

end tell

activate

repeat with v from 1 to 1000
 copy false to the end of issku
end repeat

-- open the name translation file
choosenametranslation()
--set tempname2 to choose folder with prompt "Location For Name Translation Table"
--set filenameofimage to (tempname2 as string) & "Name Translation"
--set imfile to open for access file filenameofimage with write permission
```

Script Of: application "DB Tool Combos v2.3 source"

```
--set eof imfile to 0

-- re-open the category file so I can read it in line by line
set maincatlist to {}
set subcatlist to {}
set detailcatlist to {}
set detailcatassoc to {}
set dummylist to {}

-- open the category file and read in
set catfile to choose file with prompt "Locate Categories File"
close access catfile

-- initalize the detailcatlist to a list of lists of lists
-- ASSUMES 50 SUB-CATEGORIES
-- GENERATES 200 DETAIL CATEGORIES
--repeat with v from 1 to 20
-- copy dummydetaillist to the end of detailcatlist
--end repeat

open window "Category Loading"
-- re-open the category file so I can read it in line by line
set currentdetailoffset to 0
set dbfile to open for access catfile
set v to true
repeat while (v = true)
 try
  set readcatlist to {}
  set readcatlist to (read catfile as {text} using delimiters {tab} before {return})
  if (item 2 of readcatlist = "0") then
   copy (item 4 of readcatlist) to end of maincatlist
   copy dummylist to the end of subcatlist
   copy dummylist to the end of detailcatassoc
  else if (item 3 of readcatlist = "0") then
   copy (item 4 of readcatlist) to end of item (item 1 of readcatlist) of subcatlist
   set currentdetailoffset to currentdetailoffset + 1
   copy currentdetailoffset to end of item (item 1 of readcatlist) of detailcatassoc
   copy dummylist to the end of detailcatlist
  else
   copy (item 4 of readcatlist) to end of item (currentdetailoffset) of detailcatlist
  end if
 on error
  set v to false
 end try
end repeat
```

Script Of: application "DB Tool Combos v2.3 source"

close window "Category Loading"

```
--set tempcnt to 1
-- This is debug code to test category stuff
--set tempdebugname to new file with prompt "Create Book Ticket File"
--set btfile to open for access tempdebugname with write permission
--set eof of btfile to 0
--repeat with v from 1 to length of maincatlist
-- set writestr to "Main Category (" & tempcnt & ")  = " & item (v) of maincatlist & return
-- write writestr to btfile
-- set tempcnt to tempcnt + 1
-- set writestr to "Sub(s) = " & return
-- write writestr to btfile
-- repeat with q from 1 to length of item (v) of subcatlist
-- set writestr to tab & item (q) of item (v) of subcatlist & return
-- write writestr to btfile

-- copy item (item (q) of item (v) of detailcatassoc) of detailcatlist to templist
-- set mark2 to (length of templist) as text
-- set writestr to tab & "Detail (" & mark2 & ") = " & return
--  write writestr to btfile
-- repeat with q from 1 to mark2
--  set writestr to tab & tab & item (q) of templist & return
--  write writestr to btfile
-- end repeat

-- set writestr to return
-- write writestr to btfile
-- end repeat
-- set writestr to return & return
-- write writestr to btfile
--end repeat

--repeat with v from 1 to length of detailcatlist
-- set mark2 to (length of item (v) of detailcatlist) as text
-- set writestr to tab & "Detail (" & mark2 & ") = " & return
-- write writestr to btfile
-- repeat with q from 1 to mark2
-- set writestr to tab & item (q) of item (v) of detailcatlist & return
-- write writestr to btfile
-- end repeat
-- set writestr to return
-- write writestr to btfile
--end repeat

--close access btfile
```

Script Of: application "DB Tool Combos v2.3 source"

```
set maincatlist to {"None"} & maincatlist
close access catfile

set writestr to ""
open window "QuarkXPress Converter"

on capitalize(sb)
 set cwords to number of words in sb
 set sbNew to ""
 if cwords < 1 then return sbNew
 set ch to upper(character 1 of word 1 of sb)
 set cch to number of characters in word 1 of sb
 if cch > 1 then
  set sbNew to ch & characters 2 thru -1 of word 1 of sb
 else
  set sbNew to ch
 end if
 set i to 2
 repeat while i ≤ cwords
  if word i of sb is not in {"a", "the", "and", "an", "for", "of", "to", "in", "with", "on"} then
   set ch to upper(character 1 of word i of sb)
   set cch to number of characters in word i of sb
   if cch > 1 then
    set sbNew to sbNew & " " & ch & characters 2 thru -1 of word i of sb
    else
    set sbNew to sbNew & " " & ch
    end if
  else
   set sbNew to sbNew & " " & word i of sb
  end if
  set i to i + 1
 end repeat
 return sbNew
end capitalize

on upper(ch)
 set charnum to ASCII number ch
 if charnum ≥ (ASCII number "a") and charnum ≤ (ASCII number "z") then
  set theResult to charnum - 32
  return ASCII character of theResult
 end if
 return ch
end upper
```

Script Of: application "DB Tool Combos v2.3 source"

```
on charTrim(chIn, sb)
 set sbNew to (sb as string)
 repeat while last character of sbNew as string is chIn
  set sbNew to characters 1 through -2 of sbNew as string
 end repeat
 return sbNew
end charTrim


on charReplace(chIn, sb, chOut)
 set sbNew to ""
 repeat with i in sb
  if (i as string) is (chIn as string) then
   set sbNew to sbNew & chOut
  else
   set sbNew to sbNew & i
  end if
 end repeat
 return sbNew
end charReplace

on charFilter(chLower, chUpper, sb)
 set sbNew to ""
 set chLower to ASCII number chLower
 set chUpper to ASCII number chUpper
 set cch to length of sb
 set i to 1
 repeat while i ≤ cch
  set ch to (item i of sb)
  set charnum to ASCII number ch
  if ((charnum ≥ (chLower)) and (charnum ≤ (chUpper))) then
   set sbNew to sbNew & ch
   if ch is " " then
    set i to i + 1
    repeat while i ≤ cch and item i of sb is " "
     set i to i + 1
    end repeat
    set i to i - 1
   end if
  else if charnum = (ASCII number "é") then
   set sbNew to sbNew & "&eacute;"
  else if charnum = (ASCII number "®") then
   set sbNew to sbNew & "&reg;"
  else if charnum = (ASCII number "™") then
   set sbNew to sbNew & "&trade;"
  else if charnum = (ASCII number "“") then
```

Script Of: application "DB Tool Combos v2.3 source"

```
 set sbNew to sbNew & (ASCII character 34)
 --set sbNew to sbNew & "&ldquo;"
else if charnum = (ASCII number "”") then
 set sbNew to sbNew & (ASCII character 34)
 --set sbNew to sbNew & "&rdquo;"
else if charnum = (ASCII number "‘") then
 --set sbNew to sbNew & "&lsquo;"
 set sbNew to sbNew & "'"
else if charnum = (ASCII number "’") then
 --set sbNew to sbNew & "&rsquo;"
 set sbNew to sbNew & "'"
else if charnum = (ASCII number "©") then
 set sbNew to sbNew & "&copy;"
else if charnum = (ASCII number "¢") then
 set sbNew to sbNew & "&cent;"
else if charnum = (ASCII number "°") then
 set sbNew to sbNew & "&deg;"
else if (charnum = (ASCII number "/")) then
 if i < cch then
  set chNext to (item (i + 1) of sb)
  set charnumNext to ASCII number chNext
  if charnumNext = (ASCII number "2") then
   set chLast to last character of sbNew
   set charnumLast to (ASCII number chLast)
   if charnumLast = (ASCII number "1") then
    set sbNew to characters 1 through -2 of sbNew
    set sbNew to sbNew & "&frac12;"
     set i to i + 1
    end if
  else if charnumNext = (ASCII number "4") and sbNew is not "" then
   set chLast to last character of sbNew
   set charnumLast to (ASCII number chLast)
   if charnumLast = (ASCII number "1") then
    if i > 2 then
    set sbNew to characters 1 through -2 of sbNew
     else
     set sbNew to ""
     end if
    set sbNew to sbNew & "&frac14;"
     set i to i + 1
   else if charnumLast = (ASCII number "3") then
    if i > 2 then
    set sbNew to characters 1 through -2 of sbNew
     else
     set sbNew to ""
    end if
```

Script Of: application "DB Tool Combos v2.3 source"

```
    set sbNew to sbNew & "&frac34;"
     set i to i + 1
     else
     set sbNew to sbNew & "/"
     end if
   else if i > 2 then
    set chLast to last character of sbNew
     set j to 1
    set sbNewTemp to ""
    set cchNew to length of sbNew
    repeat while j < cchNew
     set sbNewTemp to sbNewTemp & (item j of sbNew)
      set j to j + 1
     end repeat
    set sbNewTemp to sbNewTemp & "-"
    set sbNewTemp to sbNewTemp & chLast
    set sbNew to sbNewTemp
    set sbNew to sbNew & "/"
    else
    set sbNew to sbNew & "/"
    end if
   else
    set sbNew to sbNew & "/"
   end if
  else if sbNew is not "" then
   if last character of sbNew is not " " then
    set sbNew to sbNew & " "
   end if
  end if
  set i to i + 1
 end repeat
 return sbNew
end charFilter

on OLDcharFilter(chLower, chUpper, sb)
 set sbNew to ""
 set chLower to ASCII number chLower
 set chUpper to ASCII number chUpper
 set cch to length of sb
 set i to 1
 repeat while i ≤ cch
  set ch to (item i of sb)
  set charnum to ASCII number ch
  if ((charnum ≥ (chLower)) and (charnum ≤ (chUpper))) or ¬
   (charnum = (ASCII number "é")) or ¬
   (charnum = (ASCII number "®")) or ¬
```

Script Of: application "DB Tool Combos v2.3 source"

```
  (charnum = (ASCII number "™")) then
  set sbNew to sbNew & ch
  if ch is " " then
   set i to i + 1
   repeat while i ≤ cch and item i of sb is " "
    set i to i + 1
   end repeat
  else
   set i to i + 1
  end if
 else
  if (charnum = (ASCII number "/")) then
   if i > 2 then
   set chLast to last character of sbNew
    set j to 1
   set sbNewTemp to ""
   set cchNew to length of sbNew
   repeat while j < cchNew
    set sbNewTemp to sbNewTemp & (item j of sbNew)
     set j to j + 1
    end repeat
   set sbNewTemp to sbNewTemp & "-"
   set sbNewTemp to sbNewTemp & chLast
   set sbNew to sbNewTemp
   end if
   set sbNew to sbNew & "/"
   set i to i + 1
  else
   set sbNew to sbNew & " "
   set i to i + 1
  end if
 end if
end repeat
return sbNew
end OLDcharFilter

on charFilterFraction(chLower, chUpper, sb)
 set sbNew to ""
 set chLower to ASCII number chLower
 set chUpper to ASCII number chUpper
 set cch to length of sb
 set i to 1
 repeat while i ≤ cch
  set ch to (item i of sb)
  set charnum to ASCII number ch
  if (charnum ≥ (chLower)) and (charnum ≤ (chUpper)) then
```

Script Of: application "DB Tool Combos v2.3 source"

```
  set sbNew to sbNew & ch
  if ch is " " then
   set i to i + 1
   repeat while i ≤ cch and item i of sb is " "
    set i to i + 1
   end repeat
  else
   set i to i + 1
  end if
 else
  set sbNew to sbNew & "/"
  set i to i + 1
 end if
end repeat
return sbNew
end charFilterFraction

on makecombos()

tell application "Microsoft Excel"
 Activate
 tell Worksheet "Skus2"
  Activate

  set openrow to 1
  set notfound to true
  repeat while notfound
   set cellstring to ("r" & openrow as text) & "c1"
   set cellc to (the Value of Cell cellstring)
   if (cellc = 0) then
    set notfound to false
   else
    set openrow to openrow + 1
   end if
  end repeat
 end tell
end tell

set pasterows to (number of items of stylelist) * ¬
 (number of items of colorlist) * ¬
 (number of items of sizelist)

set gooddatarow to openrow
tell application "Microsoft Excel"
 Activate
 tell Worksheet "Skus2"
```

Script Of: application "DB Tool Combos v2.3 source"

```
  Activate

  -- Copy the "applied" row
  set cellstring to ("A2:AE2")
  set test to Select Range cellstring
  copy test

  -- Paste it
  --set cellstring to ("A" & (openrow + pasterows) as text)
  set cellstring to ("A" & gooddatarow as text) & ":AE" & ((gooddatarow + pasterows - 1) as text)
  Select Range cellstring
  Paste

 end tell
end tell

repeat with a from 1 to number of items of stylelist
 repeat with b from 1 to number of items of colorlist
  repeat with c from 1 to number of items of sizelist

  set aSKUrow2 to {0, 0, 0}
  set item 1 of aSKUrow2 to item a of stylelist
  set item 2 of aSKUrow2 to item b of colorlist
  set item 3 of aSKUrow2 to item c of sizelist

  tell application "Microsoft Excel"
    Activate
   tell Worksheet "Skus2"
     Activate

    -- Copy the "applied" row
    --set cellstring to ("A" & gooddatarow as text) & ":AE" & (gooddatarow as text)
    --set test to Select Range cellstring
     --copy test

     -- Paste it
    --set cellstring to ("A" & openrow as text)
    --Select Range cellstring
     --P a s t e

    -- Set the additional data
   set cellstring to ("r" & openrow as text) & "c3:r" & (openrow as text) & "c3"
   if ((item 1 of aSKUrow2) ≠ 0) then
    set the Value of Range cellstring to (item 1 of aSKUrow2)
    end if
```

Script Of: application "DB Tool Combos v2.3 source"

```
    set cellstring to ("r" & openrow as text) & "c4:r" & (openrow as text) & "c4"
    if ((item 2 of aSKUrow2) ≠ 0) then
     set the Value of Range cellstring to (item 2 of aSKUrow2)
     end if

    set cellstring to ("r" & openrow as text) & "c5:r" & (openrow as text) & "c5"
    if ((item 3 of aSKUrow2) ≠ 0) then
     set the Value of Range cellstring to (item 3 of aSKUrow2)
     end if

    end tell
   end tell
   set openrow to openrow + 1
  end repeat
 end repeat
end repeat

-- tell application "Microsoft Excel"
--  Activate
-- tell Worksheet "Skus2"
--   Activate
-- set rangestring to ("A" & gooddatarow as text) & ":AE" & (gooddatarow as text)
-- tell Range rangestring
--    Delete
--  end tell
-- end tell
-- end tell

activate

end makecombos

on resetSKU()
 tell application "Microsoft Excel"
  tell Worksheet "Skus"
   set notfound to true
   repeat while notfound
    set cellstring to ("r" & currentskunumber as text) & "c1"
    set cellc to (the Value of Cell cellstring)
    if (cellc = 0) then
     set notfound to false
     else
     set currentskunumber to currentskunumber + 1
     end if
   end repeat
  end tell
```

Script Of: application "DB Tool Combos v2.3 source"

```
 end tell
end resetSKU

on choosenametranslation()

 set query to display dialog ¬
  "Select Name Translation File" buttons {"New File", "Existing File", "Cancel"} default button 1

 if button returned of query = "Existing File" then
  -- if here, prompt the user to name a Book Ticket file to open
  set tempname to choose file with prompt "Locate Existing Name Translation File"
  set imfile to open for access file tempname with write permission
 else
  set tempname to new file with prompt "Create Name Translation File"
  set fnamestr to tempname as string
  set imfile to open for access file fnamestr with write permission
  set eof imfile to 0
 end if

end choosenametranslation


on chosen theObj -- a menu item has been chosen
 copy name of window of theObj to theWindow
 copy name of theObj to theMenuItem
 copy title of menu of theObj to theMenu

 if theMenu is "File" then
  if theMenuItem = "Quit" then

   -- commented out per betsy 2/4/99
   -- tell application "Microsoft Excel"
    --  Activate
   -- set save_file_name to GetSaveAsFilename
   --  SaveCopy ActiveWorkbook In save_file_name
    --   Close
    --  end tell
    --  activate

   close access my imfile
    quit
  end if
 end if

end chosen
```

Script Of: push button "quitButton" of window "Image Converter"

```
on hilited theObj
 quit
end hilited
```

Script Of: push button "batchButton" of window "Image Converter"

```
on hilited theObj
 set enabled of (push button "nextButton" of window "Image Converter") to false
 set enabled of (push button "skipButton" of window "Image Converter") to false
 set enabled of (push button "batchButton" of window "Image Converter") to false
 batchProcess()
end hilited
```

Script Of: push button "skipButton" of window "Image Converter"

```
global nameTable
on hilited theObj
 openNextFile(nameTable)
end hilited
```

Script Of: push button "nextButton" of window "Image Converter"

```
global nameTable
global outFileName
global inFileRef
global cropFileRef
on hilited theObj

 tell application "Adobe Photoshop® 5.0"
  activate
  with timeout of 1000 seconds
   do script "change mode to RGB"
   my savePICT(outFileName)
   my closeImage()
  end timeout
 end tell
 open for access {cropFileRef} with write permission
 set eof_value to get eof cropFileRef
 write outFileName & return to {cropFileRef} starting at (eof_value + 1)
 close access {cropFileRef}
 openNextFile(nameTable)
end hilited
```

Script Of: push button "thumbButton" of window "Image Converter"

```
global outFileName
global doneFileRef
global inFileRef
on hilited theObj

 tell application "Adobe Photoshop® 5.0"
  activate
  do script "change mode to RGB"
  do script "Thumbnail Image"
  my saveGIF(outFileName)
  my closeImage()
  open {inFileRef}
 end tell
 open for access {doneFileRef} with write permission
 set eof_value to get eof doneFileRef
 write outFileName & return to {doneFileRef} starting at (eof_value + 1)
 close access {doneFileRef}
end hilited
```

Script Of: application "CropFeed 1.3 source"

```
    end if
  end tell
end closeImage
```

Script Of: application "CropFeed 1.3 source"

```
  click button "Replace"
 end if
 if exists radio button "None" then
  click radio button "None"
  click radio button "32 bits/pixel"
  click button "OK"
 end if
end tell
end savePICT

on saveJPG(thePlace)
 tell application "PreFab Player™"
  --do menu menu item "Save As" of menu "File"
  type "s" holding shift & command
  do menu popup item "JPEG" of popup "Format"
  type {thePlace, ".jpg", enter}
  if exists button "Replace" then
   click button "Replace"
  end if
  type {jpegLevel, enter}
 end tell
end saveJPG

on saveGIF(thePlace)
 tell application "PreFab Player™"
  --do menu menu item "Save As" of menu "File"
  type "s" holding shift & command
  do menu popup item "CompuServe GIF" of popup "Format"
  type {thePlace, ".gif", enter}
  if exists button "Replace" then
   click button "Replace"
  end if
  if exists radio button "Normal" then
   click radio button "Normal"
   click button "OK"
  end if
 end tell
end saveGIF

on closeImage()
 -- close and don't save
 tell application "PreFab Player™"
  --do menu menu item "Close" of menu "File" with repeat until successful
  type "w" holding command
  if exists button "Don't Save" then
   click button "Don't Save" -- may have exported the changed version
```

Script Of: application "CropFeed 1.3 source"

```
    my saveGIF(outFileName)
    my closeImage()
    end if
    --end if
   end timeout
   end tell
   tell application "Finder" to delete {outFileRef}
   if (doneFiles does not contain outFileName) then
   open for access {doneFileRef} with write permission
   set eof_value to get eof doneFileRef
   write outFileName & return to {doneFileRef} starting at (eof_value + 1)
   close access {doneFileRef}
   end if
  on error errtxt
   if not (doneFiles contains outFileName) then
   display dialog errtxt & ": " & outFileName
   end if
  end try
 on error
  close access {cropFileRef}
  exit repeat
 end try
end repeat
activate me
beep 2
display dialog "Done processing images." buttons "OK" default button "OK"


end batchProcess

on inList(theItem, theList)
 repeat with i from 1 to count theList
  if item i of theList = theItem then return i
 end repeat
 return 0
end inList

on savePICT(thePlace)
 -- warning: if filename is maximum number of chars, the original file will be replaced
 -- if close to the max, ".gif" will be truncated
 tell application "PreFab Player™"
  --do menu menu item "Save As" of menu "File"
  type "s" holding shift & command
  do menu popup item "PICT File" of popup "Format"
  type {thePlace, enter}
  if exists button "Replace" then
```

Script Of: application "CropFeed 1.3 source"

```
  set theResult to true
 end if
 end if
end if
return theResult
end isDetailName

on batchProcess()
activate me
display dialog "Done with Cropping." & return & "About to compress and create thumbnails."

try
open for access {doneFileRef}
set doneFiles to read {doneFileRef} as list using delimiter return
close access {doneFileRef}
on error errtxt
display dialog "No previously batched files were found."
set doneFiles to {}
end try
--set filelist to (name of every file in folder (FolderToOpen as text) whose file type is "PICT") as list
--tell application "Finder" to set filelist to (files of FolderToOpen whose file type is "PICT") as list
--repeat with theFile in filelist
close access nameTable
set cropTable to open for access {cropFileRef}
repeat
 try
  set outFileName to read cropTable as text before return
  set outFileRef to a reference to file ((FolderToOpen as text) & outFileName)
   try
   tell application "Adobe Photoshop® 5.0"
     activate
    with timeout of 1000 seconds

    open {outFileRef} -- using application file id "8BIM"
      activate
     --do script "Product Page Image"
     do script "USM"
     my saveJPG(outFileName)
     my closeImage()
     if ((not my isDetailName(outFileName)) and (doneFiles does not contain outFileName)) then
      open {outFileRef} -- using application file id "8BIM"
        -- end tell
        activate
       --do script "Thumbnail Image"
       do script "thumbnail"
      do script "RGB to Indexed Color"
```

Script Of: application "CropFeed 1.3 source"

```
set inFileRef to a reference to file ((FolderToOpen as text) & inFileName)
set outFileName to read nameTable before return
repeat while croppedFiles contains outFileName

 -- display dialog croppedFiles as text
 read nameTable until ":"
 set inFileName to read nameTable as text before ":"
 set inFileRef to a reference to file ((FolderToOpen as text) & inFileName)
 set outFileName to read nameTable before return
end repeat
on error
 batchProcess()
 return
end try
set enabled of (push button "thumbButton" of window "Image Converter") to true
set enabled of (push button "nextButton" of window "Image Converter") to true
set enabled of (push button "skipButton" of window "Image Converter") to true
set enabled of (push button "batchButton" of window "Image Converter") to true
tell application "Adobe Photoshop® 5.0"
 activate
 with timeout of 10000 seconds
  try
  open {inFileRef}
  if my isDetailName(outFileName) then
    beep
   display dialog "Detail Image" with icon 1
   end if
  on error errtxt
  display dialog errtxt & ": " & inFileName
  my openNextFile(nameTable)
  end try
 end timeout
end tell

end openNextFile

on isDetailName(sb)
 set cch to length of sb
 set i to cch
 set theResult to false
 if i > 2 then
  repeat while (item i of sb ≥ 0) and (item i of sb ≤ 9) and (i > 1)
   set i to i - 1
  end repeat
  if (i < cch) and (item i of sb is "d") and (i > 1) then
   if item (i - 1) of sb is "." then
```

CropFeed 1.3 source

Script Of: application "CropFeed 1.3 source"

```
  set nameTable to open for access alias nameName
  on error
   display dialog "Error on Name Translation file" & return & "Please select another folder, or create a Name
Translation file"
   set enabled of push button "nextButton" of window "Image Converter" to false
   set enabled of push button "thumbButton" of window "Image Converter" to false
   set enabled of (push button "skipButton" of window "Image Converter") to false
   set enabled of (push button "batchButton" of window "Image Converter") to false
   return
  end try
 end try
 openNextFile(nameTable)
end getFolder

on chosen theObj
 copy name of window of theObj to theWindow
 copy name of theObj to theMenuItem
 copy name of menu of theObj to theMenu
 if index of menu of theObj = 1 then
  display dialog "Utility to Batch Process Image Files"

 else if theMenu is "File" then
  if theMenuItem is "Open Folder" then
   getFolder()
  else
   if theMenuItem = "Quit" then
    quit
   else
    if theMenuItem = "Preferences..." then
     display dialog "Current JPEG Compression level is: " & jpegLevel
     --open window "Preferences"
    end if
   end if
  end if
 end if

end chosen

on openNextFile(nameTable)
 try
  set enabled of (push button "thumbButton" of window "Image Converter") to false
  set enabled of (push button "nextButton" of window "Image Converter") to false
  set enabled of (push button "skipButton" of window "Image Converter") to false
  set enabled of (push button "batchButton" of window "Image Converter") to false
  read nameTable until ":"
  set inFileName to read nameTable as text before ":"
```

CropFeed 1.3 source

Script Of: application "CropFeed 1.3 source"

```
global nameTable
global nameName
global FolderToOpen
global inFileRef
global outFileName
global cropFileRef
global croppedFiles
global doneFileRef
global doneFiles
global jpegLevel

set preferences_folder to path to preferences
try
 set pref_file to open for access file ((preferences_folder as text) & "Cropfeed Prefs")
 set jpegLevel to read pref_file before return
 close access pref_file
 -- set the setting of gauge "jpegGauge" of window "Preferences" to (jpegLevel as integer)
 -- set contents of textbox "jpegValue" of window "Preferences" to jpegLevel
on error errtxt
 display dialog "Preferences file not found. Using default values."
 set jpegLevel to "2"
 -- set jpegLevel to contents of textbox "jpegValue" of window "Preferences"
end try
open window "Image Converter"
getFolder()

on getFolder()
 set FolderToOpen to choose folder with prompt "Select a folder to convert"
 try
  set cropFileRef to a reference to file ((FolderToOpen as text) & "Cropped Images")
  open for access {cropFileRef}
  set croppedFiles to read {cropFileRef} as list using delimiter return
  close access {cropFileRef}
 on error errtxt
  display dialog errtxt
  set croppedFiles to {}
  display dialog "No previously cropped files were found."
 end try
 set doneFileRef to a reference to file ((FolderToOpen as text) & "Done Images")

 try
  set nameName to ((FolderToOpen as text) & "Name Translation")
  set nameTable to open for access alias nameName
 on error errtxt
  try
   set nameName to choose file with prompt "Select Name Translation File"
```